長野大学紀要　第16巻第1・2号合併号　25－40頁　1994

キャンパス・ネットワークにおける資源の有効利用
ケーススタディー（II）　ハードウェアおよび情報資源

Efficient Utilization of Resources in a Campus Network (II)

平　岡　信　之

Nobuyuki Hiraoka

1．はじめに

1．1　本稿の目的

本稿はケーススタディー3回シリーズの第2部である。本シリーズでは筆者が本学の教育研究用情報システムの構築および管理に関わってきた経緯の中で、実施した実験的作業の概要と、その経験を通じて得た知見等を報告し、また、その知見の今後の情報システムへの活用の道について考察することを目的としている。

第1部である前稿[1]では、外部情報資源の活用と題して、広域ネットワーク(インターネット)への接続について、主として技術的な視点から報告を行った。実はインターネット自体は、そこにある情報資源を活用するという、情報受信者の立場から見た価値だけを持つ訳ではなく、逆にインターネット（の向こう側にいる人達）に向けて、情報を発信するための道具としての価値も大きいものである。従って、その後者の視点での議論を省略した前稿は、その意味では片手落ちだったとも言える。しかし、双方向チャンネルとしてインターネットを論じる場合、改めて「メディア」としての視点から、計算機ネットワークを（キャンパスネットワークも含めて）見直す必要がある。これについては、特にインターネット関係は最近とみにホットな領域で、前稿脱稿後の追報の必要のある事項も多々生じていることでもあり、場を改めて報告し直すことを、筆者の今後の課題として残しておくことにしたい。

従って、本シリーズでは、キャンパスネットワークを、(情報処理装置であるところの)計算機システム、の集合体であると考え、その前提に基いて行った作業や考察について、資源という観点から述べるものである。本稿では、そのシリーズの第2回として、主としてハードウェア資源および情報資源の活用に関して述べる。

1．2　資源とは

計算機の類のもの[1)]が世に現れて約半世紀になろうとしている。その歴史は資源確保の努力の歴史であるともいえる。当初最も高価かつ重要な資源であったCPU(中央処理装置)[2)]資源の処理能力向上（つまり単位時間あたりに得られるその資源量の向上）は、絶えず技術的課題であり続けているし、主なオペレーティング・システムが土台としているタイムシェアリングやマルチプロセスといったソフトウェア技術[3)]も、そのCPU資源の有効利用を目的として開発されたものである。

ただ、昨今の状況は、それよりもやや複雑である。「計算機」としてのコンピュータにおいてはその様相は今もさほど変わらないが、現在我々の身近にあるコンピュータ類は、情報を処理するだけでなく、蓄積し、転送し、さらにマンマシンインタラクションを行うという、八面六臂の機能を持つ機械である(ことを期待されている)。その機能毎に（互いに重複しながらも）個別の資源を要求する。例えば単体マシンの物理的ハードウェア

1）いわゆるフォン・ノイマン型アーキテクチャは基本的に今なお継承されてはいるが、その用途を考えると「計機」という“昔の名前”で呼ぶことを実はつい躊躇してしまう。ちなみに我々は会話では通常「ましん」「はーど」「のーど」等の名称で呼ぶ。

2）1チップ内に作り込まれるようになって以降のものは、特に装置そのものの呼称としてはMPU或はプロセッサ等の方が一般的であるが、資源としてその処理能力を論じる時にはやはりCPUと呼ぶことが多い。

3）ソフトウェア上の変革は次の世代のハードウェアに反映されることも多いため、これらをソフトウェア技術と言い切ってしまうことは適切でないかも知れないが。

資源に限っても、CPU以外に主記憶装置(メモリ)、補助記憶装置(ディスク等)、転送路（バス等）がそれぞれ適量必要となる。これらのいずれかが適量よりも不足した場合には、その部分がボトルネックとなって、必要とする機能を実行できないか、実行のための別のコスト(大抵の場合は余分な「時間」)を必要とするか、という結果になる。その場合、他の資源は活用されずに「遊ぶ」ことになる。なお、これらの資源の「適量」は、そのシステムの用途等の様々な状況に依存し、どこにでも通用するワイルドカード的数値は存在しない。

物理的ハードウェア資源以外に、無形の資源も考慮に入れる必要がある。計算機を動作させるためにソフトウェアが必要である[4]し、そのソフトウェアを含めたあらゆるデータは重要な情報資源として扱う必要がある。ソフトウェアに関しては、その使用権（ライセンス）も場合によっては資源としての扱いを必要とするし、また、ハードソフトを問わない論理的な資源として、例えばアドレス空間のような数字空間や名前空間も貴重な資源である場合がある。なお、(言及するのが遅れたが）本稿では「資源」という用語を、有形のもの、天然のものといった一般に使われる意味に限定しない広い解釈で、「有限のもの」「(何等かの)コストのかかっているもの」といった意味合いで使用している。これに違和感を覚える読者もいるかもしれないが、情報システムの世界の慣用語として理解して頂きたい。ちなみに、情報システムに関して、一般的な用語における「資源」に属するものとしては、設備や消費財の材料物質や、システム運用のための電力、プリンタ用紙等の消耗品があり、これも経済上、或は道義上、決して無視してはいけないものであろう。

さらに、昨今の計算機は通常は単体で置かれるのでなく、何等かの伝送線で相互接続されたネットワークシステムの一部として存在することが多い(本学の状況も然りである)[5]。その場合、ネットワークを通じて同種の資源を相互融通することが技術的に可能となり、資源の有効利用の可能性は広がる。ただし、この場合、そのネットワーク自体が資源としての性格を持つことになり、話がさらに複雑になる可能性がある。また、このネットワーク技術とは別に、異種の資源の可換性を実現する技術も存在する[6]。さらに視点を拡大して、組織内の情報システム全体をマンマシンシステムとして捉えた場合、それに関与する人間もまた資源の1つであることになる[7]。

こういった複雑な状況において、真に最適なソリューションを得ることは至難の技[8]であり、さらなる研究が必要であることは明らかではあるが、それに至る道標として、筆者の浅薄な見識と経験に基づいて整理し得る部分を、本シリーズの第3部で報告したいと考えている。本稿ではそれに先立って、前述の資源のネットワーク内相互融通と、異種の相互可換性実現を中心に、技術的な視点での報告を行う。

2．出発点

2.1　第2次システム

本学産業情報学科（以下、情報学科と略記する）の設立は88年春であるが、それに先立って、教育研究を目的とする情報システムが導入設置されている。このシステムを筆者は第2次システムと呼んでいる（但しこの呼称は、筆者の便宜上の分類であり、公的に使われているものではない)[9]。筆者が本学に赴任し、情報システムに関わり始めた（本研究に関する作業を開始した）のはその3年後であり、これが本稿の出発点となる（以下、作業開始時点と記す)。その3年間に若干の設備追加等が行われてはいるが、筆者にとっては先史の話であり、それらを一括して本稿では第2次システムとして扱う [2]。また、下記の分類も、導入時の意図や名目とは無関係に、(用途等における)

4）というより、ソフトウェアを実行することが計算機の存在意義である（筈）という方が正しいのだが。

5）実は日本ではまだこれからという状況ではあるが。

6）という大層な表現をしたが、例えば計算機を使わずに全部手作業でやっつける、というのも異種の資源の可換性実現の(わかりやすい)「テクノロジー」の1つである。

7）これらをひっくるめて組織の構造ごと再構成しようという試みが（リエンジニアリングと称して）行われつつあるようだが、残念ながら筆者の研究はそこまでは手が届かずにいる。

8）個別の状況毎に、最適解を得るためにさらに人的資源を消費することを考慮すると、これは実質的には不可能だと言えるだろうが、それに向かっての志は必要だろう。

9）使っているうちに定着するであろうことを目論んでいることは、ご推察の通りである。

その後の実態をもとに筆者が再構成したものである。なお、この第2次システムは本稿執筆時点でも基本的にその用途で継続して（筆者は第2.5次に分類しているが）利用されている。（但し、94年秋を目標に大幅更新の計画が進められており、本誌発行時点では第3次と呼ぶべきシステムが稼動を始めていると思われる。）

第2次システムは概ね以下のサブシステムから構成され、2つの建屋に跨る1本のイーサネットセグメント[10]に（部分的に）接続されている。

a）ホスト機　DEC社製スーパーミニコン（VAX8600）
　設置場所：マシンルーム（実習室隣）
　用途：集合教育（主として実習授業）

b）パソコン　NEC社製パソコン　約100台（PC9801UV；フロッピーモデル）
　設置場所：実習室、演習室
　用途：集合教育（主として実習授業）において、
　• 単体パソコンとしての利用　⇒環境1
　• a）にログインするための端末⇒環境2
　の2通りの用途。また、独習用にも利用。
　接続：イーサネット直結でなく、シリアルケーブル（RC232C）、ターミナルサーバ（Annex）を経てイーサネットに接続。

c）EWS（エンジニアリングワークステーション[11]）類　Sun等　16台
　設置場所：各研究室（主に情報システム研究所3、4階）に分散配備
　用途：ゼミにおける小人数教育と研究
　　⇒環境3

d）その他　PC（パーソナルコンピュータ）類　Mac等　約10台（メーカ、機種は様々）
　設置場所・用途：c）と同様　⇒環境4
　接続：スタンダロンが殆どだが、Macの一部はlocaltalkで相互接続

10）3種類（Thin, Thick, 及び光）、10本以上のケーブルから成るが、すべてリピータ接続であるため、機能的には1本と考えるべきものである。

11）最近は必ずしもエンジニアリングが主用途とは言えなくなっており、さらにPC（パソコン）との機能的差違も明確でなくなってきているため、「マルチユーザ向けデスクトップ機」といったあたりが正確な分類ということになるだろう。

2.2　問題と課題

情報技術の世界は、（規格動向や製品動向に限れば）1年先すら予測できない程に、流れの速い世界であり、すでに構築され稼動し利用されている（しかも自分が貢献したのでない）システムに関して、後から批評めいたことを述べるのは、単なる後知恵でしかない反則行為であるかも知れない。また、前稿でも触れたが、この第2次システムは、

- （当時まだ大・中型機やオフコンに流れがちだった大学教育の世界で）Unixを採用したこと
- 1つの設備群（上記のb））でUnixとMS-DOSの2つのプラットフォームの利用を可能にしたこと
- イーサネットの採用

といった、その先進性において評価すべき点も多々ある。が、それでも構築後3年が経過した、筆者の作業開始時点では、様々な事情（それも別途分析する必要があるかも知れないが）で、その「時代」における教育研究実施に（少なくとも筆者にとって）不都合を生じていたことは、記録に留めておいて差し支えはなかろう。

80年代から90年代にかけて、計算機利用環境に大きな変化が起こった。その主要なキーワードを挙げると、

A）ネットワーク化
B）日本語処理の普及
C）GUIの一般化

ということになるだろう。筆者は、「計算機システムの有効な利用の道を探って行こう」というようなコンセプトで授業なりゼミなりを行っているが、その場面において、その題材となる計算機システムが旧態依然のものでは、前に進むのが困難になる。少なくとも、その「時代」における、計算機利用の「スタイル」程度のものは経験により摑んでおく必要はある。ところが上記A～Cの観点から第2次システムを見た場合、それに適した環境が用意されていない。ゼミや小グループ等の、ある程度以上の人数が同時に利用可能な環境は、前述の1～3であるが、それぞれ

環境1：Bについてはパソコンの機能を用いて一応可能だが、AとCはほぼ絶望的（シリア

ルラインでは実用的な通信プロトコルは実行不可能だし、グラフィックデータについては蓄積しておける外部記憶領域がない）

環境2：Aは利用可能だが、Bは利用不可、Cは絶望的（1と同じ理由による）

環境3：Aは利用可能だが、Bは利用不可（DECもSunも米国製のマシンおよびOSで動作する）、Cは利用可能だが業界の実質的な標準の方式は使えない。

というように、いずれも不完全であることが分かる。そこで、この状態から、「時代」に取り残されていない何等かの環境が存在する状態に近づけることが、当面の課題となる。それも、経済的に極力慎ましやかなソリューションを得るということも条件となる（この条件は特に誰かが設定した訳ではなく、いわば「場の雰囲気」及び筆者の性分や、これまでフリーソフト等に親しんできた筆者の経験等から、自動的に醸し出されてしまったものだろう。）

さて、学内のどこかに（比較的）先進的な環境を展開するプランの、その対象として、とりあえず上記の環境1～4のいずれかを選ぶ必要がある。このうちの環境1、2は、集合教育用の環境であり、基本的には常に画一的である必要がある（授業時に、座った机によって使い方が違うというような環境だと教員も学生も非常に困る）し、また、環境改善作業のために機器の運用が停止することもまた望ましくない。一方の環境3、4については、統一性や運用継続性にさほどの厳しい要求はなく、また、そのため管理が行き届かなかったことも災いして、（幸か不幸か）利用度はかなり低い（実は何年も眠ったままのマシンもあった）。さらに環境3については、前述の分析に見たように、唯一「絶望」の文字のない環境であり、遊休資源を活用するという点を考えても、これが最も適した環境であると考えられる。かくして、前稿のインターネット接続と並行して、第2次システムに対して、これが筆者の当面の課題となった（これも別に誰が命令した訳でもないのだが、やはり場の雰囲気と、「必要なことは自分でやる」文化の中で育ってしまった筆者の、自然な反応だったと言えるだろう）。

3．EWS環境の改善の必要性

3．1　ハードウェアの状況

さてその環境3、即ちEWS環境だが、前述のように、主としてSunワークステーションで構成されている。具体的には、

Sun 3／50（モノクロ）10台　4MB
Sun 3／60（カラー）　4台　4～8MB
Sun 4／260（カラー）　1台　8MB,
　ディスク 500MB
News 841（モノクロ　1台　8MB,
　ディスク 250MB

という構成である[12]。その大勢を占める機種であるSun 3については、ディスク容量を記述していないが、これらに接続して並べて設置可能な、「シューボックス」と呼ばれる外形寸法のディスク装置（殆どがストリーマテープ装置付き）が合計12台ある。それぞれ容量は140MB(1台だけ280MB)である。

これらのマシンは、Sun 3同士を除いて、バイナリ（プログラム等）の互換性はない。しかし、前述のVAX 8600を含めて、これらのマシンを制御するOSは、いずれもBSD系UNIXに属するものであり、ほぼ類似の発想で管理できるものであることは幸いであった。ただし、それでも各メーカ毎の独自の拡張や方式があったり、また、OSの版による大きな差違（BSDの4.2版から派生したものと4.3版から派生したものの間の差違も大きい）もあり、一筋縄ではいかない部分もある。いずれにしても、数の点で優勢であるSun 3から、その有効活用の試みを始めることになる。

3．2　日本語表示の方式

通常、EWSはコンソール画面では日本語は表示できない。少なくとも上記の機種はそうだし、最近でも状況はさほど変わらない。EWSにおいては、マルチウィンドウの利用が前提となっており、何等かのウィンドウシステムを実現するソフトウェアを通じて、その窓に日本語を表示することができれば事足りるからである。ではそのウィ

12）いずれも当時はエンジニアの憧れの的だった機種であるが、今ではこのスペックはノートPCかと見紛う程度のものである。いや、その感覚さえもあくまで本稿執筆時点以降、すぐに陳腐化するのかも知れない。

ンドウシステムでの日本語処理はというと、上記機種では、さすがに国産機であるNewsは、すでに日本語化されていたが、Sunに関しては、当時(Sun3購入時点)標準でサポートされておらず[13]、当時の海外製ソフトウェアの多くと同様、日本語文字の表示、入力や処理が出来ないだけでなく、プログラムによっては、扱うデータの各バイトのMSB(最上位ビット)を、特に必要もなく0に落としてしまうこともあり、データの中継、仲介的な用途にも問題を生じる場合がある、という状況ではあった[14]。勿論マニュアルも当然すべて英語である。なお、これはVAXについてもほぼ同様だった。

Sun3の場合、日本語を表示させる方法として、(筆者の調べた範囲で)3つの方法があった。

1．JNIX：OSおよび純正のウィンドウシステム（Suntools）に対する追加ソフトウェア群として、サードパーティによって販売されていたもの[3]。本学でも3台分購入してあった。ただし、Sun OSのバージョン3にのみ対応している。

2．X-Window：汎用のウィンドウシステム[4]（以下、Xと略記する)。V10(ヴァージョンテン）以降、ソースコードの形で一般に（フリー）で入手可能になっており、V11になってからもリリースを重ね、当時(作業開始時点)すでにR4（リリースフォー）が出てからかなり経過していた。

3．JLE：Sun OSバージョン4（以降）の日本語拡張として標準サポートされている（ただし当時は別買だった)[5]。Xをベースにしており、Xとの相互乗り入れは可能だが、機能豊富のため、Sun3には若干重すぎるという難点がある。

上記のうち、1および3は表示に限らず、入力や操作、編集といった機能までサポートされた、「これ1つでOK」型のものであるのに対し、2は、X自体の入手および各マシンに合わせたコンパイル、インストールが必要であるばかりでなく、表示以外の部分（入力、編集および諸ツール）については別途ソフトウェアの入手が必要[15]であり、また、ベンダーによるサポートも受けられない。というデメリットもあるが、やはりここではXを中心に据えた解決策を探っていくことにした。理由としては（言わずと知れた）経済的事情以外に、

- X独自の資源融通のメカニズム(後述)は有用
- 業界の実質標準の位置を占めている
- 今のNews以外にも今後Sun以外の様々な機械が導入される可能性がある。

という点があげられる。勿論、当面のつなぎとして購入済のJNIXも使ってきているし、時期を見て予算確保をお願いして、漸次OSのバージョンアップを図るということも行っている。

3.3　日本語の入力と編集

コンピュータを使う人間を（乱暴に）大別して、

- 感性で使う人
- 論理で使う人

の2つに別けられるとすると、情報学科の目的は、後者の育成が主体であると筆者は考えている[16]。論理でコンピュータを使う限りは、キーボードと適切なエディタは必須の道具であり、それはタッチタイピングで操作されるのがふさわしい。従って、情報学科の最初の課程はタッチタイピングであるべきだし[17]、(特にエディタにおいて)タッチ

13）米国の製造社はまだ日本法人を開設しておらず、商社が製品をそのまま輸入して販売しているだけに近いものであった。

14）これらのマシンやOSの開発者にとっては、ASCIIコードでは到底表現しえない何万種類もの複雑な文字セットを利用するような民族が極東に存在していることなど、思いもよらない、といったところだったのだろう。ちなみに、最近のソフトウェアの「日米同時リース」等というニュースを目にすると、この時代の流れが恐ろしくなる。

15）最近のリリースになるほど、そういった問題は解消されてきて、Xの配布媒体中に何でも揃うようになってきた。さらに日本語入力方式などについては、提唱された複数のソフトが収められていて、今度は逆に「選ぶ」手間を生じるようになっている。などというのは贅沢な悩みであるが。

16）誤解のないように注記しておくが、昨今のいわゆるマルチメディア化の動きの中で、感性でコンピュータと関わる人の層が(情報発信者においても受信者においても)増えてくることを、筆者は否定しているわけではない。感性でコンピュータを使うための枠組みを設定したり提供したりする人は、主として論理でコンピュータに関することになるという考えである。

17）もし出来ることなら、予科のようなものでも設置して、予め習得しておければ、なお望ましいとすら思うが、勿論これは暴論であろう。

タイピングの習慣化を妨げない、「筋のいい」エディタを提供するべきであり、これを妨げるような無神経な造作のもの（即ち、ついついホームポジションから離れたファンクションキーや矢印キーを使ってしまいたくなるもの）は、未来ある学生の目からは極力隠蔽するべきである、というのが筆者の基本的な考えである。

その「筋のいい」エディタとして、Unix の世界では、vi と Emacs [6]（およびそれらの類似品や派生品）がある[18]。この両者の優劣をここで論じることは避けるが[19]、1点だけ根本的な差違をここで挙げておく。

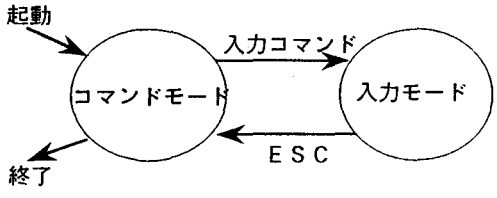
vi：一種の modeful editor；以下の2つのモードを持つ（第1図）。

テキストモード：ユーザのキー操作は入力テキスト(即ちデータ)として受理される。

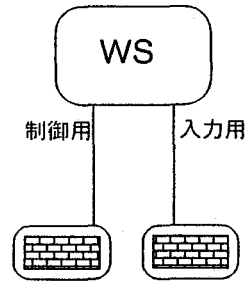
コマンドモード：ユーザのキー操作は何等かの編集操作の指示と解釈される。

Emacs：一種の modeless editor ； 常に以下の2つの操作が可能。通常のキー操作は常に入力テキストとして受理される。編集操作の指示等のためには、Ctrl、Meta等の修飾キーにより修飾されたキー入力（およびその組み合わせ）を用いる。

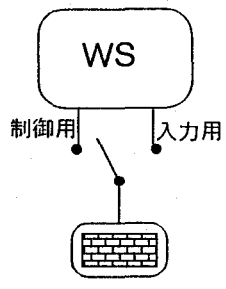
この違いは、表現を換えれば、第2図（a）のようにデータ用及び制御用の2台のキーボードを持つシステム（これだと混乱は少ないが、勿論極めて使いにくいだろう）を、（b）のように1台のキーボードとして実現する場合の、スイッチの違いとして考えることも可能である[20]。



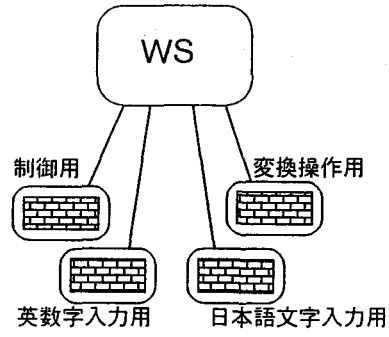
第1図　vi のモード遷移図



第2図(a)
2種類のキーボード



第2図(b)
仮想的切り替え



第3図　4種類のキーボード

さて日本語の入力編集を行う場合を考えると、入力すべきデータは（単純化すると[21]）日本語文字と英数字の2種類になる。それに加えて、変換中（勿論カナ漢字変換方式であることを前提とする）の文節群に関しての操作を行うキーボードも必要となり、従って、第2図（a）の代わりに第3図のような4台のキーボードを持つシステムを想定する必要があることになる。これを1台にまとめるようなスイッチ（に相当するキー操作）を設計するのは、なかなか大変な作業である。

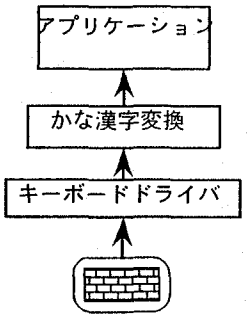
が、実はそれ以前に、カナ漢字変換（クライアント）プログラムの動作場所に起因する制限の方

18）実際、これらは親切なメニューやガイドは通常は出してくれず、慣れないものにとっては敷居の高いツールである。が、新入生の時期(恐らく最もテンションの高い時期）にこれを乗り越えるパワーを持っていないとすれば、それ以降に遭遇する学業上の（或はそれに限らない）どんな壁も乗り切れないのではないかと思うのだが。

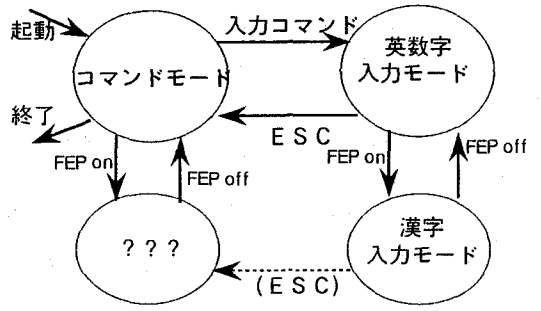
19）ネットニュース等ではたびたび vi 対 Emacs 論争が起こり、大抵は個人の趣味や主観に基いた水かけ論に陥ることが多いので、最近では「宗教論争」扱いされている。

20）データ用チャネルと制御用チャネルを1本の（論理的な）線上に多重化する際の技術的問題点は、コミュニケーションのあらゆる場面で生じ得る。例えば口述筆記して（もらって）いる時に発せられた「じゃなくて」という語句はどう解釈する（される）か、というような問題とも共通している。

21）実際には、英数字およびカタカナは1バイト文字（いわゆる半角）と2バイト文字（いわゆる全角）の2種類があり、話は複雑なのだが、ここではあまり本質的でないので省略する。



第4図　フロントエンドプロセッサ



第5図　FEP付きviのモード遷移

が大きい。通常、OS（を構成するプログラム群）を日本語化する場合、その各プログラムにカナ漢字変換機能を組み込むということは行わず、変換機能つきウィンドウ、或は変換機能つきシェル、というように、殆どのアプリケーション（応用プログラム；以下、アプリと略記する）が共通に利用する要衝部分に変換機能を組み込む（開発や管理のコストを考えれば当然のことだし、通常はこれで充分である）。この場合、ユーザのキー操作により入力された文字コードは、第4図のように、いくつかのプロセスを通過してアプリ（この場合はvi）に届けられることになる。ユーザから見て、カナ漢字変換のプロセスが、アプリよりも手前側にあるため、この形のソフトをフロントエンドプロセッサ（略してFEP）と呼ぶこともあり、さらにはカナ漢字変換プログラムそのものを総称してFEP（フェップと読む）と呼ぶコミュニティもある[22]。ここでは誤解が生じる必配はなさそうなのでその呼称を借用する。英数字入力時には、そのFEPは完全に透明な状態だが、日本語文字入力時にはFEPが入力列を一時預り[23]し、変換を行い、確定の後、アプリに送られる。

従って、そのFEP型のカナ漢字変換方式とviエディタを組み合わせた場合、第1図に代わるモード図は、第5図のようになる。ただしカナ漢字変換操作中のモード遷移（FEPにより違う）はこの図には組み入れていない。ここで問題となるのが、左下の「？？？」モード（名前がつけられない[24]）である。これは第3図のどの鍵盤にも対応しない。このモードが、「次に（viが）入力モードに移る時には自動的に日本語文字入力モードに移る（或は、最近の入力モードでのFEPの状態を記憶している）ことを除いては、通常のコマンドモードと同等に振る舞う」というようなモードであれば、さほど害はないのだが、実際には、ここではviのコマンドは使えない。これは独立して動く2つのプロセスのそれぞれ2つのモードの掛け合わせの結果生まれてしまう邪魔者、でしかなく、操作を著しく繁雑にし、ユーザを混乱させる。「？？？」モードには簡単に落ちてしまう反面、「？？？」モードから日本語文字モードへの移行には、順序正しく3つの操作を行うことを必要とする（しかも殆どのFEPのオン・オフ操作は同一のキーによるトグル動作であるため、それぞれ正しく1回ずつ行う必要がある）。FEPの種類によっては、ESCキーは素通りさせるものもあり[25]、その場合は図の点線のように日本語文字入力モードから「？？？」への方向のみ1操作で移行するという非対称性をもつことになり、ユーザの混乱をさらに増長する。コンピュータ嫌いの育成を目的とするのでなければ、このFEP（型カナ漢字変換プログラム）＋vi」の組み合わせは避けるべきだろう。

となると、有望なのはEmacsということになる。

22）このFEPという呼称には批判もあるようだが、パソコンの世界などではこれで定着してしまったような感がある。勿論和製英語である。というよりも、そもそも1バイトを越える寸法の文字コードセットを使ったり、その入力に通常の英数字鍵盤を使ったり、という発想からして、すべて和製なのだが。

23）OSによっては「横取り」と表現する方がふさわしいメカニズムで動いている場合もある。

24）計算機の動作等を擬人化して表現するような文化に浸ってきている筆者などがこれを無理に命名したりすると、放送コード（の類のもの）に引っ掛かってしまいそうで恐いので避けた。

25）実は前述の（2.1節のb）パソコンに標準装備させたFEPもそのタイプであった。

幸い、「FEP＋Emacs」では、上記のような致命的な落とし穴はないし、第3図のキーボードの仮想的な重ね合わせという見地からも、さほどの不整合はない。ただし、使い勝手の改善において、方式そのもの（即ちFEP方式でカナ漢字変換を行うこと）に起因する限界はある。ユーザの意志を正確かつ速やかに伝達するための言語としてキー操作（列）を考えた場合、ユーザは（互いに状況伝達を行う手段を持たない）2つのプログラムを相手にしなければならないからである。この解決策としては、

- カナ漢字変換プログラムをアプリに内蔵させる
- アプリとFEP間の伝達規約を定め、双方に実装する

といった手段が可能であり、現在ではすでにそういった方向での研究・開発の成果は盛んに公表されているが[26]、当時（作業開始時点）この方向のもので我々が容易に入手可能だったものとしては、Wnn（後述）のサーバを利用するクライアント機能をEmacs[27]に組み込むための拡張としてEGGなるものが知られているのみだった。いずれにしても、ネットワーク上で日本語をタッチタイピングで使う、という目的のためには、エディタとしてはEmacsしか考えられないという状況だった訳である。

以上は、基礎教育を行う視点から見たEmacsの必要性に関する議論であったが、実は、Emacsを導入する理由は、それだけには終わらない、何といってもEmacsはエディタと分類するのが馬鹿げているほどの拡張性を持ったシステムであり、Emacs上に用意された数多くの（勿論原則としてフリーの）ツール群の使用を通じて、計算機の用途の無限の可能性（の少なくとも一部は）を垣間見ることができるし、また、実際、朝から晩まで[28]Emacsの中から出ることなしに、自分の仕事のすべてを行っている技術者も数多くいる[29]ほど、身体に馴染みやすいシステムなのである。いずれの分野においても、良いもの、優れたものに触れる機会というものは、専門家を育成していく過程においては必要性の高いものであろう。但し、残念ながらEmacsはかなり重い（主としてメモリ空間とCPU資源を派手に消費する）プログラムであり、第2次システムのVAX8600は、集合教育の人数分（例えば30人以上）のプロセスを動かせるパワーは持っていないし、学内の他所からその資源を融通することも不可能である。当面は小人数用として使うしかない。

さらに難点をあげると、上記のいずれのソフトウェアも、資源の「大食漢」である。特にディスク領域について顕著である。例えばEmacsの場合、ソースだけで約10MBを消費とする。それをコンパイルし、実行可能な状態にするためには、さらに約10MBを必要とする。X-windowに至っては、それよりも1桁以上も大きな領域を要求する。これらは、日本語を扱うための特殊事情（辞書やフォント情報が大きな領域を占有する）によるものであったり、多機能化に伴う様々なライブラリや関連ソフトウェアが添付されることに起因するものであったりする。いずれにしても、これが90年代の計算機ソフトウェアの傾向である。

4．ディスク領域の問題

上記の方向で整備を進めていくために、そのままでは、ディスクに関して2つの点で絶対的資源不足の問題が生じる。1つは、ディスク装置の数が少ないこと。少なくとも2台の本体に対して、ディスクが接続できないことになる。もう1つは、空き領域の量が少ないことである。140MBのスペースに、OSをインストールして、スワップ領域（仮想記憶領域のうち、実記憶からはみ出た部分を保持するディスク領域）や作業領域など、定常的運用に必要な領域を確保すると、それだけで約100MB強を消費する。と、ユーザ用空間は40MBしかない。これでは、前述のソフト整備の作業にも足りない。当時のディスクはまだ高価だったため[30]、これは小額の投資では解決しない問題である。そ

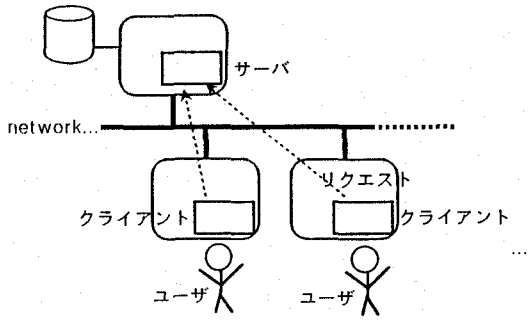
26）かといってEmacsを不要にするものには、まだお目にかかってはいないが。

27）正確に言うと、Emacsの日本語対応版（NEmacs）又は他国語対応版（Mule）である。

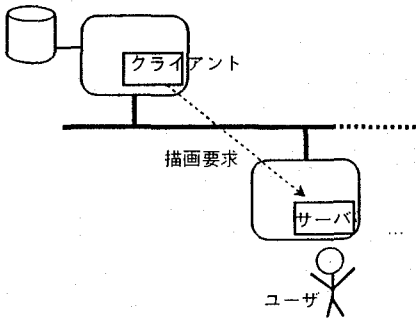
28）晩から朝まで、という場合も勿論ある。

29）実は筆者も（Emacs＋EGGがちゃんと動く職場にいた時は）そうだった。

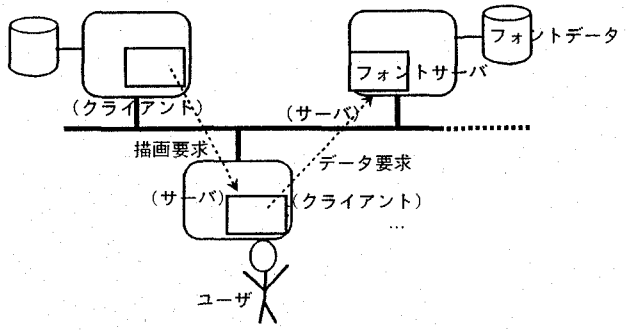
30）大量生産によるディスク装置の価格低下が始まったばかりで、メガ1万（1MBあたり約1万円）を割り始めた位の頃である。



第6図(a)　クライアントサーバ型ソフトウェア



第6図(b)　X-window の場合のクライアントとサーバの関係



第6図(c)　サーバでもありクライアントでもあるソフトウェア

こで、本章の各項に述べるように、極力「ソフト的」な解決を図ることとした。

4.1　クライアントサーバ型ソフトウェア[31)]

一般的にクライアントサーバ型（以下C／S 型と略記する）のソフトウェアは、第6図（a）のように2種類（クライアント側にはさらに複数のものが存在する場合がある）のソフトウェアからなり、ネットワークを通じて両者が連携を行いながら、ユーザの要求する機能を実行する。但し、Xのようにユーザインタフェースを司るソフトウェアの場合、に関しては、第6図（b）のように、ユーザから見てクライアント・サーバ両プログラムの位置関係が逆転している場合があり、また、第6図（c）のようにフォントサーバを利用する場合、各クライアントソフトからの描画要求等に応えるXサーバが、サーバであると同時に、フォントサーバに対してデータを要求するクライアントとしても動作するという複雑な関係も生じる。

いずれにしても、これらのC／S 型のソフトウェアを積極的に活用することにより、ディスクスペースへの要求が緩和できる。例えばカナ漢字変換ソフトウェアである Wnn [7] は、その主要機能を実現するサーバはネットワーク内に1つ動いておれば足りる（勿論、性能的要求等により、複数台による運用を行うこともできるが）ため、サーバプログラムや辞書等のデータを、全マシンにインストールする必要がない訳である。また、ネットニュースを読むための（ニュースリーダと呼ばれる種類の）ソフトウェアについても、NNTPというプロトコルに対応した版（のクライアント側ソフト）と、それに応えるサーバを用意することにより、ニュース記事の（前稿でも触れたが膨大な量の）データを、複数のマシンに保持しておく必要がなくなった。その他、現時点では導入が未完了であるが、システムのログ管理や、電子メールの読み書きについても、C／S 型運用への移行

31）実際にはネットワーク上で動作するソフトウェアは殆どが内部的にはC／S 型の動作をしているのだが、通常の Unix に標準装備のもので、ユーザが（クライアントを）起動すればすぐ使えるものや、本稿のテーマである資源の有効利用への直接の関係の薄いものについては、ここでは触れない。

により、ディスクスペースや管理上の手間の軽減が期待される。

4.2　ディスクレス運用

前述の、ディスク装置の数の問題は、一部のEWSについてディスクレス運用を行うことにより解決することにした。このときディスクレスEWSは、ネットワーク内の[32]他機のディスク領域の一部を「間借り」することになる[33]。勿論、「間貸し」する側のマシンで、予めそのための受入れ準備（領域の確保やサーバプログラムの起動）は必要である。通常、ディスクレス機は、自分のMACアドレス（ネットワーク上での物理アドレス；ROMの中に保持されている）以外の情報を持たないため、ブート時には、このMACアドレスを手掛かりにしてIPアドレスを決定したり、さらにソフトウェア（ブートコードやOSそのもの）をダウンロードしたりという操作を行うための要求をネットワーク上に送信する。ディスクレス機をサポートする親機は、領域の間貸し以外に、これらの要求にも応える必要がある。

領域の「間貸し」機能は、Sun OSにおいては、バージョン3ではnd（Network Disk）というプログラムで提供され、バージョン4では、NFS（後述）機能の中に統合されて提供される。後者の方が、管理の容易さや柔軟性において優れており、また、サーバ側（のマシンまたはソフト）が短時間でもダウンすると、前者ではクライアント（ディスクレス機）も無条件にダウンしてしまうのに対して、後者ではダウン中にクライアント機のI／O要求が発生しなければ継続運用可能であるという機能的な差違も大きい。しかし、ここでは一挙にOSをバージョンアップすることは控え、両者を混合する形で、ディスクレス機の運用を行っていくことにした。なお、実際には、次章に述べる理由から、1部のマシン（主にSun 3／60）にディスク装置を各2台ずつ接続する一方で、殆どのSun 3／50を（不足分の2機だけでなく）ディスクレス運用することになった。

32）あらゆるマシンは電源に接続すると同時にネットワークに接続する、ということを前提としている。

33）この機能は大抵のEWSが持っている。また、PCにおいてもハードウェアの追加により利用できることが最近わかった。

4.3　ファイルの共有

Unixのユニークな[34]機能の1つとして、「マウント」がある。1台のマシンに接続した複数のディスク装置（或はそれを区分したパーティション）について、それぞれの内部が木構造のファイルシステムとして組織化されており、それらの木構造同士を適当な点（一方の根と一方の葉）で接続（マウント）することにより、仮想的に単一の木構造を構成するというものである。これにより、Unixでは例えば「ドライブ」という概念なしに、ディスク資源や情報資源にアクセスできる。さらにネットワーク時代のUnixでは「リモートマウント」が可能であり、ネットワークで接続された別機のファイルシステムをもマウントし接続することにより、複数のマシンに跨った（或はネットワーク全体に広がった）単一のファイルシステムを構成できる。その機能の代表的なものが、Sunにより提供されるNFSである[8]。これを用いると、情報の共有が容易であり、また、ディスク領域も節約できる。その手順は、概ね以下の通りであった（第7図）。

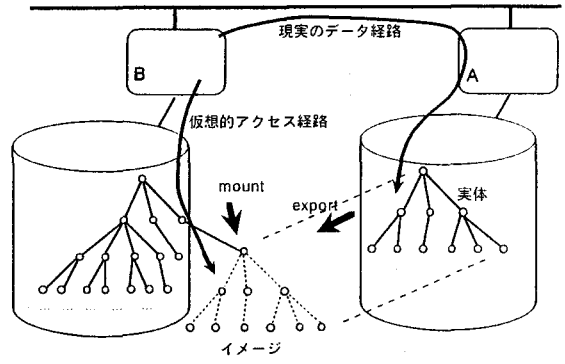
1）共有すべきファイル（或はファイル群）をマシンAに置く。

2）そのファイル群（を含む木構造）をexport（他機からのマウントを許可）する。

3）マシンB、C、…からその木構造をマウントする。

4）このファイルがA、B、C…から利用できる。

なお、Sun OS（のバージョン4以降）には、類似の（或は関連した）機能として、RFSやTFSも



第7図　NFSマウントの仕組み

34）Unixおよびその亜流の普及により、今やユニークでもなんでもないあたりまえの機能になっているのだが。

用意されており、NFSとの併用も可能ではあるが、

- NFSが実質的に業界標準になっている
- 複数の方式の併用による管理の複雑化は避けたい。
- NFSに関る諸問題は後述の方策により解決可能
- TFSやRFSを必要とする局面に（これまでの所）遭遇してこなかった[35]

という理由により、動作実験を行うにとどめ、導入はしていない。

OSを構成するファイル(やディレクトリ)は、大別すると以下の2種類に別れる。

a）参照のみ行う（追加や変更を行わない）ファイル

b）参照、追加、変更を行うファイル

さらにbは、その追加、変更を行う主体が何であるかによって、3つに区分することができ、これらは最近のUnixでは以下のようにファイルシステムのサブツリー（或はディスク装置のパーティション）に別けて置かれる慣例になっている。

a）／usr　：(追加、変更なし)
b1）／　　：システム管理者が変更
b2）／var　：デーモン（バックグラウンドで動作またはスタンバイしているプログラム）
b3）／home：一般ユーザ

このうち、aとb3は共有が可能である。b1とb2については、マシン毎に独立した領域を持つ必要があり、これにスワップ領域を加えて約30～50MB程度（章頭で述べた必要量の2～3分の1）が、マシン1台あたりの必要領域ということになる。なお、4.2BSD以前のUnix（Sun OSではバージョン3以前）では上記の区分が未分化であるため、OSのインストール後に、ファイルの移動やシンボリックリンクの作成などの、手作業の手間は必要であった。

35）例えば1つのソースファイル（またはソースディレクトリ）から複数の種類のマシン用の実行ファイルを作成する場合などに、TFSは有用だとされているが、シンボリックリンクや新しいmakeプログラムの活用により代替できる。ただ、RFSにあってNFSにない機能（デバイスファイルの共有）を利用したい局面が1度あった。それはプリンタ（を扱うプログラム）の共有が目的であったが、そのためだけにRFSを導入するのも大層であるので、代替として利用できる軽いソフトウェアを探すことにして、これについては保留してある。

本章で述べた種々の工夫の併用により、資源が揃わずに「眼って」いたマシンを「起こす」ことから始まって、優れたフリーソフトを中心とした、OS標準添付外のソフトウェアの導入による環境改善を行うためのディスク領域を確保する所まで、実現することができた。勿論、これらのソフトウェアも、どこか1箇所にインストールしたものをNFSにより各マシンで共有することができ、これにより導入作業の簡略化も行える。また、あるマシンのCPU資源がタイトである場合などに、一部の作業を別の（ヒマな）マシンに移転して実行させる（一種のプロセスマイグレーション[36]）ことも、これにより容易になる。

5．管理上の諸問題

5．1　NFSの利点と限界

5.1.1　データ共有における得失

NFSを用いてのデータ共有について前章で述べた。実際には、NFS以前から、Unixにはftpやrcpといったデータ転送ツール（及びプロトコル）が実装されており、これらを用いてデータを共有することは可能ではあったのだが、これらに比べて、NFSを使った場合は、以下のような得失を持つ。

△1　転送先にディスク領域を必要としない（仮想的にディスクが存在するように見せかける技術であるため）

△2　ユーザが明示的に転送の指示を出す必要がない。

△3　1つのデータに対して、ネットワーク内に複数のコピーを作る必要がない（複数のコピーの一貫性を維持できない、或は維持するための管理コストが大変になる、という心配がない）

▼1　プロトコルの性質上、転送効率はやや低く、その分、ネットへの負荷も大きい

▼2　要求がある度にデータが転送されるため、ネットに負荷をかける

36）ただし手動での移転であり、これは誇大表現かも知れない。5.1.1で述べるように、ユーザが資源（ここではCPU）の位置を意識しないでいい環境が理想であり、現在、プロセスマイグレーションと題して行われている研究は、それを目指しての、「プロセスの自動(であることを当然とした）マイグレーション」である。

▼3　信頼性の低いネット（例えば広域網）での頑強性に欠ける

従って、ネットワークに帯域幅や信頼度の問題がなければ、この長所の部分が発揮される訳である。

特にEWSやPCを用いた分散型作業環境においては、諸資源が分散して配備されてしまい、全体として複雑なシステムになってしまう傾向がある。その諸資源を利用するために、ユーザが様々な工夫（資源の配置を調べたり覚えたり指示したり）を必要とするようでは、資源の有効利用は望めない。理想的には、ネットワーク内の（分散した）全資源が、ユーザから見ると単一の計算機であるかのように見えること（位置透過性）が望まれる。NFS（およびその同類のファイルシステム）は、ディスク領域および情報資源に関して、それを（不完全ながらも）実現する技術であると言える。

5.1.2　C／S型システムとしてのNFS

以上に述べたように、NFSは非常に有用な機能である。NFS自身もC／S型の動作をしており[37]、Unixの基本機能のレベルでの情報共有を可能にするため、NFSさえあれば他のネットワークソフトは不要であるかのように思えるほどである。しかし、実際には、以下のように、機能的および性能的な限界が存在する。

1）排他制御の問題：1つのファイルに対する読み／書きのリクエストが複数のクライアントから同時に発生する場合、NFS自身ではそれを適切に処理できない場合がある。

2）効率の問題：例えばマシンSが保持する大きなファイルFの中からある小さな情報断片Iを抽出してマシンCで使いたい場合、NFSに頼った場合、ファイルF全体をSからCに転送する（検索はCで行う）必要があり、S内で検索を行う場合に比べて必要な転送量が大きく、従って全体として非効率的である。

3）NFS自身が依存する情報の共有：例えば他のマシンのアドレス情報などの情報は、NFS自身の起動に必要であり、従って、NFSで共有することはできない。このためのソフトウェアとしてNIS（後述）がある。

従って、適宜、他の（それぞれの用途に適したプロトコルを持つ）C／S型ソフトウェアとの併用を図っていくことが必要である。

5．2　運用上の問題

教育、研究の場において、資源は豊かであるに越したことはない。マシンの数は多いほうがいいし、できればその種類も豊富である方が望ましい。また、マシンを利用しての教育・研究に、マシン（やソフト）を使うまでの準備期間は短いほうがいい[38]。というのがユーザの立場であるが、マシン管理、ネットワーク管理の立場からすると、マシンの数や種類が多いことは、大きな悩みの種である。管理上の手間は、概してマシンの数および種類に比例するのだが、それ以外に、運用において以下のような問題が浮かび上がって来た。

5.2.1　パス名の一貫性の欠如

NFSにおけるマウントの指示は、クライアントである各マシン毎に、その管理者が自由に設定できる。これは利点（柔軟性）でもあるが、その結果、マシン毎に、ある情報資源にアクセスするためのパス名が違ってくることになる[39]。参照すべきファイル名をプログラム内にハードコーディングされたソフトウェアの場合、必要なファイルが見つからずに、実行ができないことになるし、第一、複数のマシンを共有している（時によって空いているマシンを使う）ユーザは、混乱する。

5.2.2　共有可能範囲の多様性

情報資源として共有可能なものでも、その共有範囲が種類によって違う。本学の環境では、概ね以下のように分類できる。

1）学内ネット全体（PCやMacも含む）

37）但し、NFSはUnixの基本機能と（見かけ上）同等の機能を提供し、あまりにUnixに同化してしまっているため、C／S型アプリケーションとは見なされていないようである。

38）勿論、その準備期間そのものも、教育や研究としての価値はあるし、筆者のゼミでもそれを重視して来ている。が、皆が皆それにかかずらわされる必要はないだろう。

39）話のぶり返しになるが、RFSを利用した場合にはこの問題は生じない（RFSではマウント点の集中管理を行うため）。しかし、今後Sun以外のマシンの導入の可能性もあり、すべてのUnixに実装されている訳ではない技術を使うことは、危険が大きいと考え、RFSへの移行はしなかった。

2）Unix
3）同一バイトオーダのマシン（Sun と News など）
4）同一メーカ機（Sun など）
5）同一アーキテクチャ機（Sun 3）
6）同一OSバージョン（Sun OS バージョン4のマシン）

情報の種類によって、適当な範囲での共有が望ましいが、厳密にこれに従って共有方式を設計するのは、徒らに問題を複雑化させるだけであり、当面は6）を主体として、バイナリ（実行形式のプログラム）の共有を図り、徐々に広い共有範囲の情報資源用の領域を準備していくことになるだろう。

5.2.3　システムとしての信頼性の低下

ネットワーク上でのサービスでは、サーバ（のマシンおよびソフト）が稼動していなければ、クライアントはその機能を利用することができない。NFS でマウントして利用しているファイルが、そのクライアント機が動作するために必要なファイルである場合、サーバのダウンが即ちクライアントのダウンにつながることになる。1台の故障が、複数台のマシンに影響を与えることになり、その結果、システム全体としての安定稼働率が低下する訳である。

5.2.4　依存関係のループ

Unix マシンは（例えば NFS に関して）クライアントであると同時にサーバであることができる[40]。前述のクライアントはサーバに依存している訳だが、マシンの起動やシャットダウンの際には、その依存関係を考慮して、立ち上げはサーバを先に行い、クライアントから順に落としていくようにしないと、途中で引っ掛かって二進も三進もいかなくなる場合がある。さらにその依存関係が複雑に交錯し、ループになってしまうと、厄介である。

5.2.5　安全性との両立

例えば/etc/termcap というファイルがある。様々な端末の特性定義情報を収めたファイルであり、画面への表示の際に、様々なプログラムがこのファイルを参照する、Unix では基本的なファイルの1つである。新たな種類の端末（または擬似端末ソフト）が導入されると、それに応じた定義情報を（望むらくはネット内の全マシンの）/etc/termcap ファイルに追加する必要がある。かといって、これを NFS で共有すると、マシンがトラブルに陥った際の管理作業（普通 NFS マウントせずに行う）において、例えば編集作業等に不都合が生じる。便利さと安全性との両立が困難となるケースである。

5.3　解決策

上記の問題は、以下のような方策で解決を図っている[41]。

5.3.1　ディスクの偏在化

例えば各マシンに接続した 140 MB の領域の、半分をprivate(そのマシン専用)に、半分をpublic（ネット内で共用）にすることは、上記5.2.4の問題を引き起こしやすい。そこで、前章に述べたように、ディスクを2台ずつ[42]ペアにして、各Sun3（主に性能のやや高いモデル60の方）に接続し、残ったマシンをディスクレスにすることにより、主にサーバとして振る舞うマシンと、クライアント専門のマシンとに区別するようにした。この副作用として、ディスクレス機の電源は安易に切断する[43]ことができるようにもなった[44]。

5.3.2　自動マウントの利用

NFS マウントは、通常は静的に行われる。即ち、マシン（クライアント）の起動時に一括してマウント動作を行い、そのまま常時マウントしたまま

40）最近の用語でいう、peer to peer 型の LAN である。

41）すでに稼動している環境への変更等を伴うため、全マシン一斉にエイヤアで実施という訳にはいかず、今なお完了しておらず、問題は多数残っている。

42）SCSI の場合は機能的に7台まで接続可能ではあるが、そうした場合には SCSI バス（でなければ CPU かメモリ）がボトルネックになることが予想されるため、2台というのが適当な線であろう。

43）ディスク装備機の電源の頻繁な上げ下げは、ディスクの寿命を縮めるとされており、サーバ機能のフルタイム動作（夜間でも、自宅等から電話回線経由で利用したり、外部ネットとの継続的通信を行ったり、というニーズはあるので）を可能にするためにも、（電力は浪費になるが）連続運転が普通になっている。

44）とはいえ、電力資源に関しては些細な改善でしかない。格段の電力大食漢であるVAX8600の運用時間が短縮していないから。

マシンが運用されるものである。これに対して、動的なマウントおよびアンマウント（マウントの解除）を行うプログラムがある。SunOSバージョン4以降に付属するautomountがそれである。これは、デーモンとして動作するプログラムで、リモートのファイルやディレクトリを参照する際に、(マウントされていなければ)自動的にマウントを行い、そのパスの利用が一定時間途切れれば、自動的にアンマウントを行う。これを用いることにより、5.2.4 の問題はほぼ完全に回避できる。

なお、automountと同等（今なお改良が行われており、その結果、実際にはautomountよりも高機能になっている）の機能を持つ amd（Auto Mount Deamonの略）というフリーソフトもあり、これはSun OSバージョン3でも動作する。そのため、現在のところは、マシンによってautomountとamdを併用しているが、両者の管理方式の違いが管理を複雑にする恐れもあり、今後はamdに移行していくことを考えている。

5.3.3　複数サーバの（選択的）マウント

automount では（以下特に書かないがamdでも同様である）あるパス名にマウント可能なサーバを複数指定することができ、その場合、automountはその中から適当なサーバ（通常は最も早くレスポンスを返したマシン）を選んでマウントする。従って、サーバ候補のうちの1台以上が動作しておればマウントは成功する。これにより、5.2.3 の信頼性の問題は軽減することができる。

5.3.4　NISの利用

前述（5.1.2の3）のようにNFS共有できない情報として、ホスト名とIPアドレスの変換表や、ユーザ名のリスト（単体マシンでは/etc/hosts および/etc/passwdに保持される）などがある。これらを集中管理し、各マシンから利用するための、一種の分散データベースとして、SunではNISというシステムが用意されている。勿論、管理の労力軽減の道具として、筆者も積極的にこれを利用している。

automount（およびamd）は、このNISにより管理されたマップ（マウントを行うパス名やサーバ名の表）情報により、動作の指示を与えることができる。集中管理された（従って基本的に同一内容の）マップを各マシンが利用することにより、同一の情報に対してアクセスするためのパス名の一貫性が維持され、5.1.1 で述べた問題は回避できる。また、マップの書き方の工夫により、単一のマップでも、マウントする側のアーキテクチャやOSの種類を判断して、違う種類の実体を選んでマウントさせることもできる。例えば、

/usr/local/bin として、
　クライアントがSun 3 Os 3 なら
　　server : /pub/local/bin. sun 3os 3 を
　クライアントがSun 3 Os 4 なら
　　server : /pub/local/bin. sun 3os 4 を
　選んでマウントする

というようなことが可能となり、5.2.2 で述べたアーキテクチャやOSによる共有範囲の限定も比較的容易に行える。

5.3.5　共有と分配の併用

前述（5.2.5）の/etc/termcapなどは、共有するのでなく、分配するのが適切であろう。つまり、いずれかのマシンをマスターに設定し、マスター機上の、このファイルを、ネット内の各マシンに、適宜（定期的に、或はマスターに変更があった際に）コピーする、という方式である。このコピー操作を自動的に行うためのツールとしては、cron, rdist, make 等が考えられ、コピー実行のタイミングに関する方針によって、選ぶことができる。特にrdistは、この目的のために用意された、細やかな制御が可能なコマンドである[45]。

また、/etc/termcap以外のファイルにおいても、下記の条件を満たす場合には、NFSによる共有でなく、分配を行う方が適していると考えられる。

1）ディスク領域に余裕がある
2）ネットワークが高負荷の傾向がある
3）そのファイルに対して全体の管理者以外の者（一般ユーザ等）による変更が発生しない

6．まとめ

6．1　結果

以上に述べたように、NFSをはじめとするネットワークソフトウェアを組み合わせて利用するこ

45）このあたり(およびそれ以降)で未来形で記述した部分は、検討だけ行って、まだ実施を行っていない部分である。

とにより、ネットワーク内で資源を共有または相互融通し、時代に追従した環境改善を進めることが可能になった。この間にディスクの価格は急速に低下し、（円高も相まって）現時点でギガ（バイトあたり）10万（円）の時代に入っており、3章で述べた程度の領域要求に対しては「購入」という安易な（かつ価格的に無理のない）解決は可能になっているが、そうなればなったで、昨今は音声や映像といった、「資源大食漢」ともいえるデータタイプの利用が広がりつつあり、ここで蓄積した技術の必要性は決して消失しない、一種のいたちごっこは続いている。

ここで用いた技術は、概ね以下のような技術である。

1）リモート（離れた所）のCPUを動かす技術（リモートログイン等）
2）その結果をローカル（自分の前）の画面に表示し、ローカルで実行しているかのように見せかける技術（X）
3）リモートのディスク領域や情報資源をローカル側にあるかのように見せかける技術（NFS）
4）リモートで行われているサービスをローカル側から利用する技術（C／S型ソフトウェア）
5）それらの情報を集中管理する技術（NIS等）

OSに付属して提供されたものと、フリーウェアとして（一部はサードベンダによる有償品として）流通しているものを組み合わせてこれらを実現している。本研究において新しい要素技術やソフトウェアを開発することは行っていないが、計算機システムの、エンドユーザに近い場において必要な技術として、すでに世の中に存在するものを、取捨選択し、組み合わせて利用する技術を蓄積していくことは必要であり、本稿ではその試みの一部を報告したことになる。

6．2 今後の課題

この試みは、何よりも人的資源の貧弱な環境下で行っており、従って、その環境改善にしても遅々とした歩みであるといっていいし、本稿で省略した（次稿での報告を予定している）事項、及び今後取り組むべき課題として、概ね以下のようなことが残されていると考えている。

6.2.1 周辺機器の共有

本学ネットワーク内でディスクと並んで稀少資源であったのが、プリンタである。これについては、次稿で述べる。さらに昨今ではCD-ROMドライブ、スキャナ等、周辺機器の種類が増えてきており、これらも有効に利用すべきであろう。これら可搬媒体への入出力を行う機器の場合には、無闇に広範囲にわたる共有を行うことは問題が多く、共有と占有の適度な併用が必要であろう。

6.2.2 PC類との資源共有

本稿ではUnix及びその標準的ネットワークプロトコルであるTCP／IPに基づく技術に限定して述べたが、実際のキャンパスネットワークにはそれ以外のマシンも存在し、特にPCの類のものが、数の上で大勢を占めるのが一般的傾向であり、本学でも同様である。これらのPC類もまた、ネットワークに接続することにより、資源共有に参加できる。ただし、ネットワーク機能を標準で持たない機種や、別のプロトコル体系のサービスや通信媒体が標準的に利用される機種が多く、これらの混成ネットワークの中での資源共有には様々な困難が存在する。

6.2.3 時計の管理

複数のマシン間での資源共有を行った場合、そのマシン相互間でそれぞれの内蔵の時計（の示す時刻）が不一致である場合に、ソフトウェアによっては不都合を起こす場合がある。TCP／IPではネットワークを通じて時計を調整する方式が用意されており、またインターネット内には、その基準として参照できるマシンも運用されているので、これを利用するとともに、上記6.2.2の対象マシンに関しても適用していく道を探る必要がある。

6.2.4 セキュリティ

ネットワークの活用が広がると共に、セキュリティの問題も発生しやすくなる。それぞれのソフトウェアがセキュリティホールになる可能性を秘めているし、ネットワーク上を様々な重要情報が流れるようになると、その漏洩の可能性もゼロではない。かといって、これらを恐れて情報流通の

道を閉ざしたり、使い勝手を極端に低下させるような処置は、大学として行うべきではない。適当なレベルでセキュリティ対策を導入していくのが適切であろう。

6.2.5　技術の普及

パソコンそのもの、或はその拡張としてマルチメディア機器の類のものは、価格低下も相まって普及が進んでいるが、それを使いこなす技術（プログラム製品でない意味での「ソフトウェア」）は、なかなか普及しないのが実情である。資源共有はある意味では機器製造業者等にとって有難くない技術である訳で、少なくとも広範な宣伝は行われていないし、また、単体パソコン利用に慣れた一般ユーザには、ネットワークを利用するという発想がなかなか伝わらない傾向がある。これを広く伝えていくことも研究者の責務の1つだろう。本稿もその試みの一環ではあるが、何か読者のヒントにでもなれば、幸いである。

6.3　謝辞

前稿と重複するため個別のお名前は失礼ながら省略させて頂くが、日頃指導、助言及び協力を頂く、本学、特に情報学科の先輩教員方に感謝する。また本学92年度特殊研究助成金は前稿と同様に本稿の内容にも役立たせて頂いたこと、さらにそれ以外にも、情報システムの改善に必要不可欠な諸資源（主としてハードウェア）に関する予算手当等において理解と協力を頂いている、各年度の学部長はじめ本学運営にかかわる方々にも感謝する。

また、本研究は本学学生に人並の（他の大学に対して遜色ない程度の）情報システム利用経験を提供したいという筆者の動機にも多分に基づいており、その対象として（残念ながら少数だが）好奇心旺盛な学生の存在（および要求）が、大きな支えになっていることにも言及しておきたい。中でも和田ゼミ出入りの2名の（元、および現役の）学生、栗俣氏および竹下君については、要求するだけでなく自らも環境改善にコントリビュートしてきてくれたことに対し、異例ではあるがここに名前をあげて感謝を申し述べる。

（ひらおか　のぶゆき　講師）

（1994.7.23　受理）

参考文献[46)]

[1] 平岡信之「キャンパス・ネットワークにおける資源の有効利用(Ⅰ)」長野大学紀要第15巻4号、'94

[2] bit別冊「知のキャンパス」共立出版、'89、pp.163-168

[3] CSK「JNIXマニュアル」、'87

[4] 斎藤信男（監）「Xウィンドウと GUI」ソフト・リサーチ・センター、'91

[5] 日本サン・マイクロシステムズ「JLE概要」、'91

[6] R. Stallman（竹内ほか訳）「GNU Emacs マニュアル」共立出版、'88

[7] KABA「Wnn＋GMW入門」岩波書店、'90

[8] H. Stern（倉骨訳）「NFS and NIS」アスキー出版局、'92

46）この業務において筆者が用いた主たる情報源はネットユースおよび関連ソフトウェアのドキュメント（或は場合によってソースプログラム）だったりするため、それらを逐一ここに列記することはもはや不可能であり、また、本稿で扱った分野について、これから勉強し始められる向きには、よく整理された書物（例えば[7]はシステム構築者の鼠右の書としてお薦めする）や雑誌記事等が利用可能になっている（だからネットニュースが不要になる訳ではないのだが）いるため、ここでは省略した。